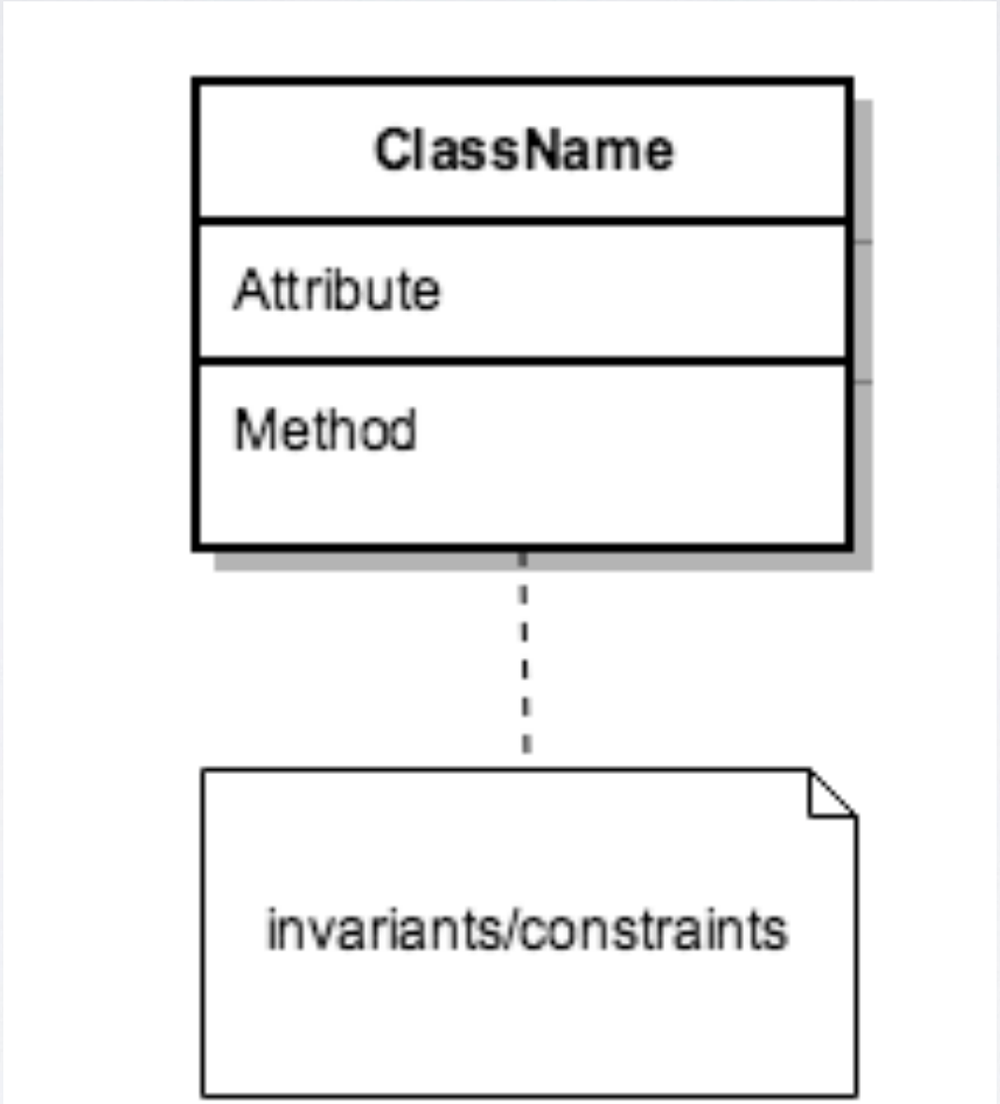
UML CLASS DIAGRAMS

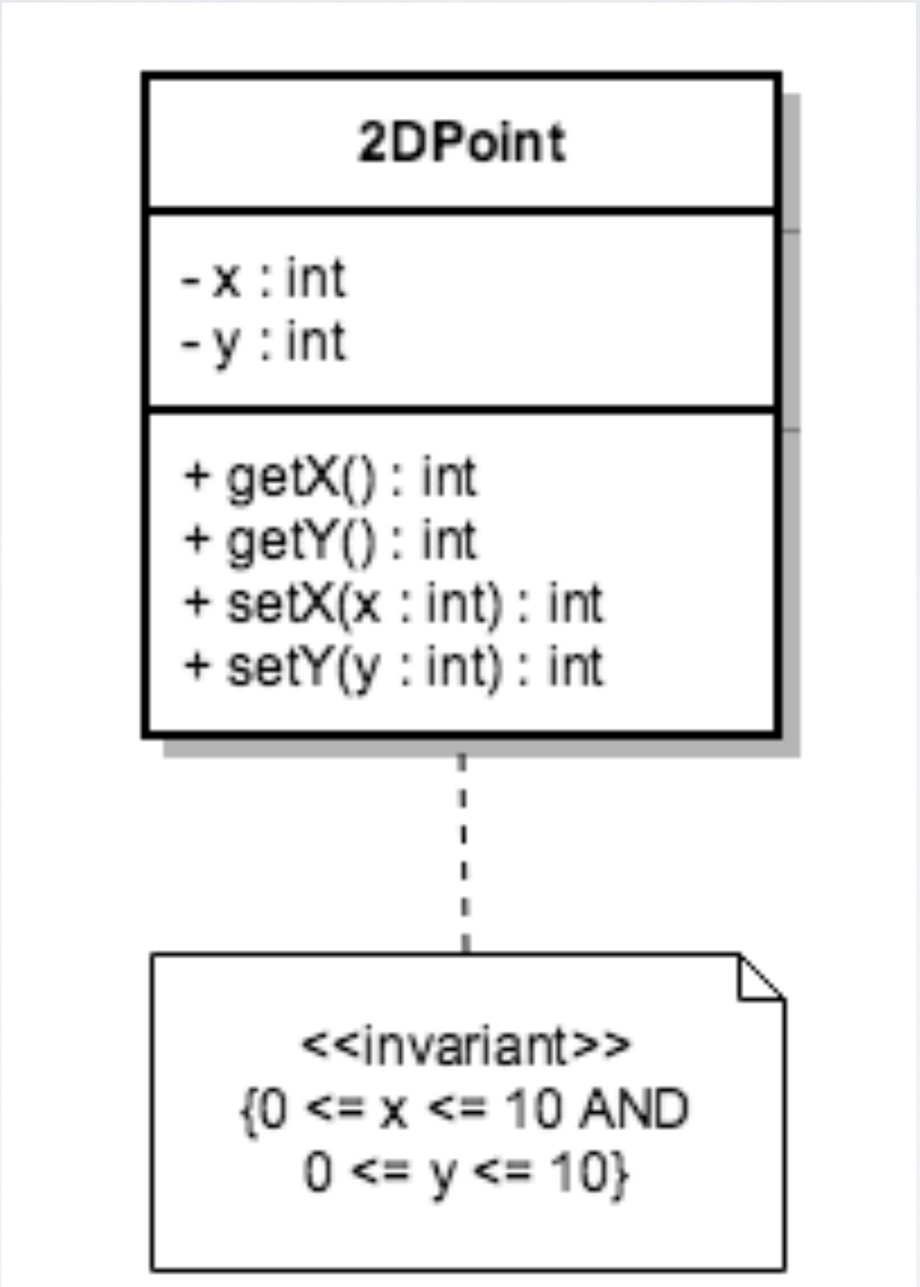
CAC430

Slides adapted from Dr. Eugene Syriani, Universite de Montreal

CLASS DIAGRAM



EXAMPLE


2DPoint
- x : int
- y : int
+ getX() : int
+ getY() : int
+ setX(x : int) : int
+ setY(y : int) : int
<<invariant>>
{0 <= x <= 10 AND
0 <= y <= 10}

ATTRIBUTE VS. VARIABLES

- Attribute - represents abstractly defined property, independent of its internal implementation (defined at design-time)

- Variable - internal implementation mechanism (defined at implementation-time)

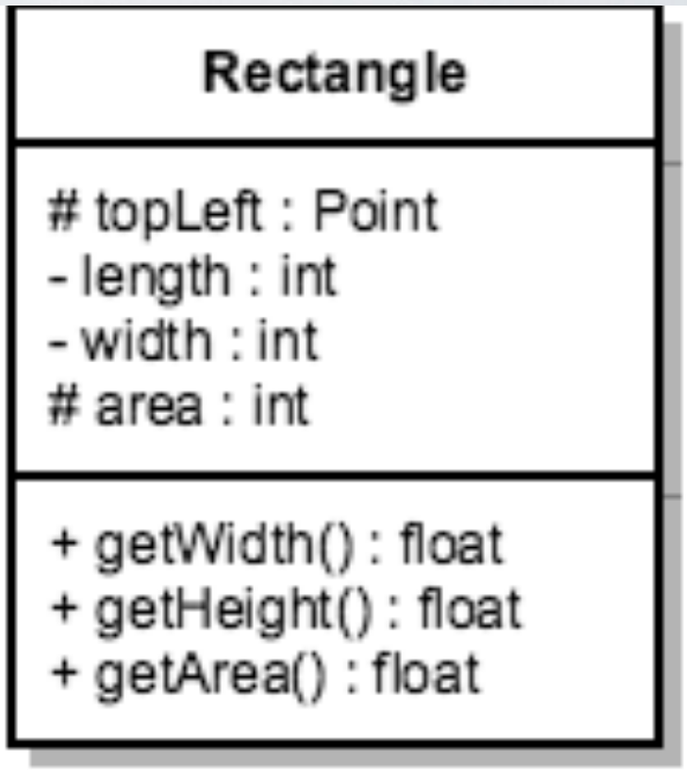
- Attributes may be implemented as variables

  - Could be a series of variables for one attribute

  - One attribute could be composed of other attributes

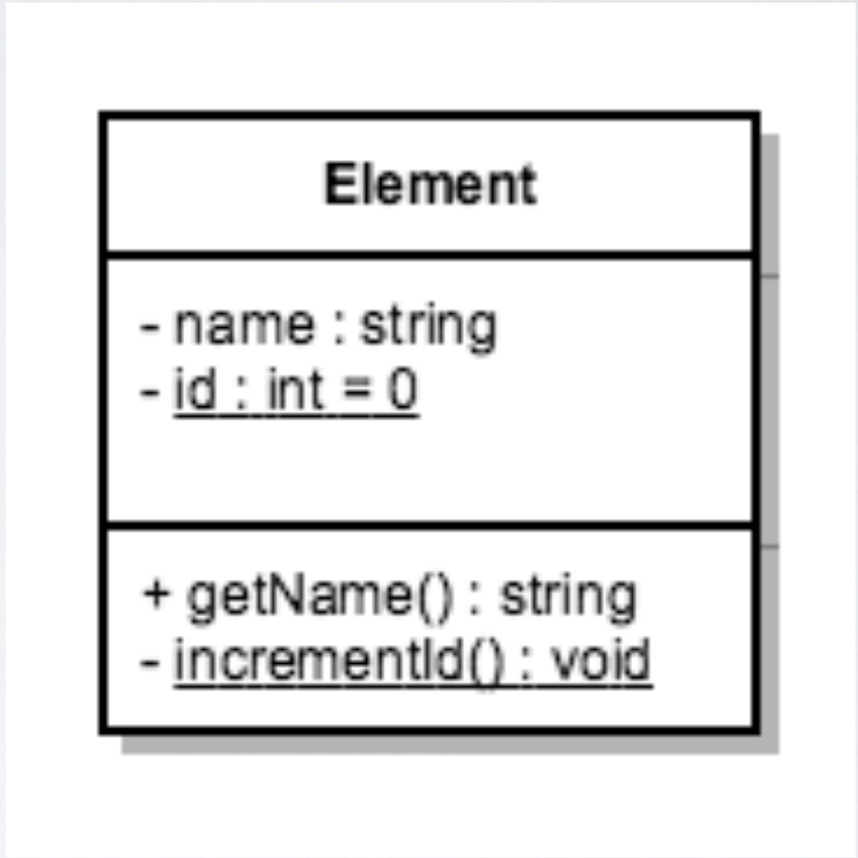
VISIBILITY

- + {public}: visible to any class
- # {protected}: visible to class and its subclasses
- - {private}: visible only to the class
- ~ {package}: visible to any class within the package



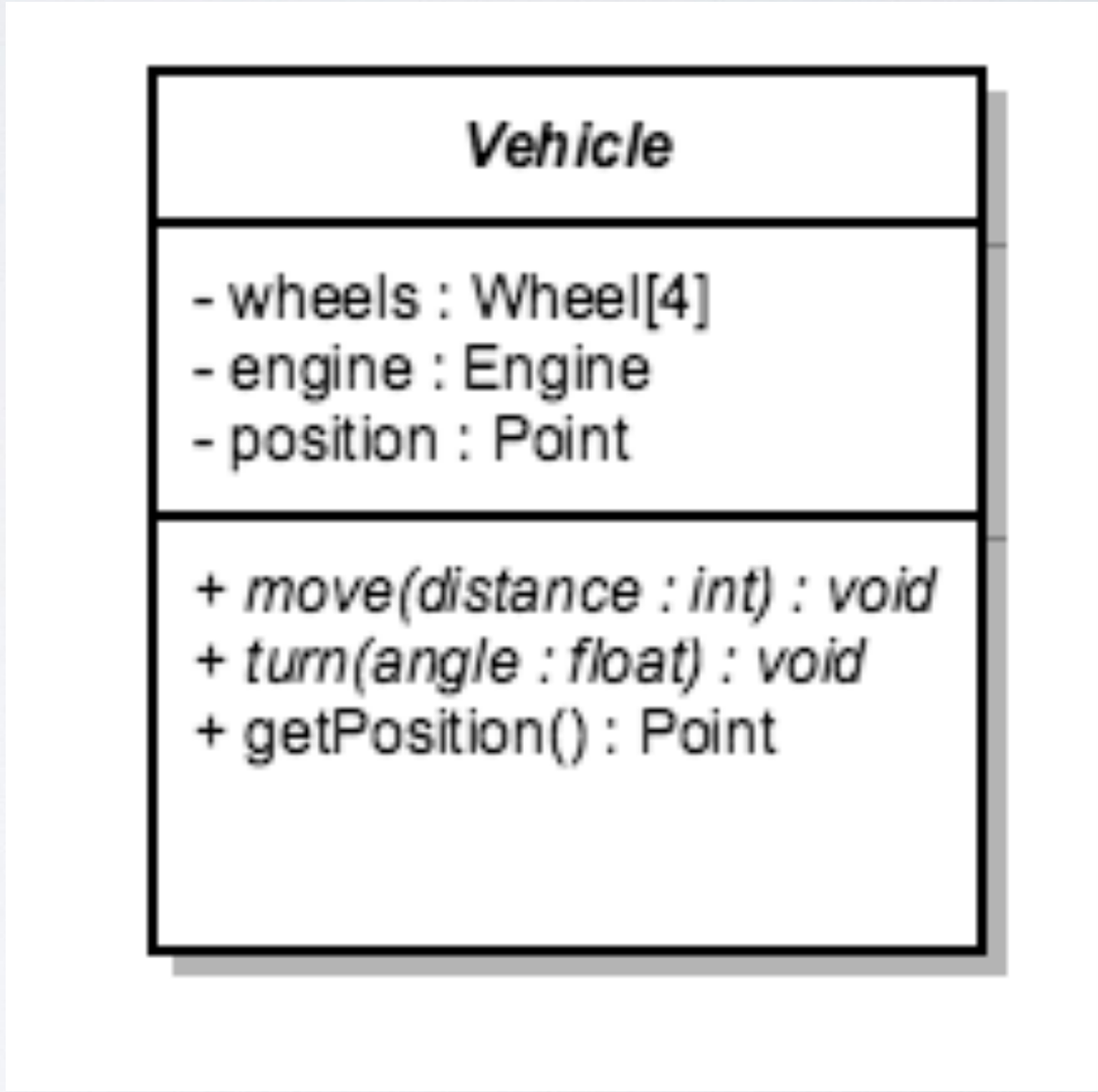
STATIC FEATURES

- Class attribute - all instances share the same attribute
  - ID generator
- Class methods - operation applied to entire class, depends on static attributes



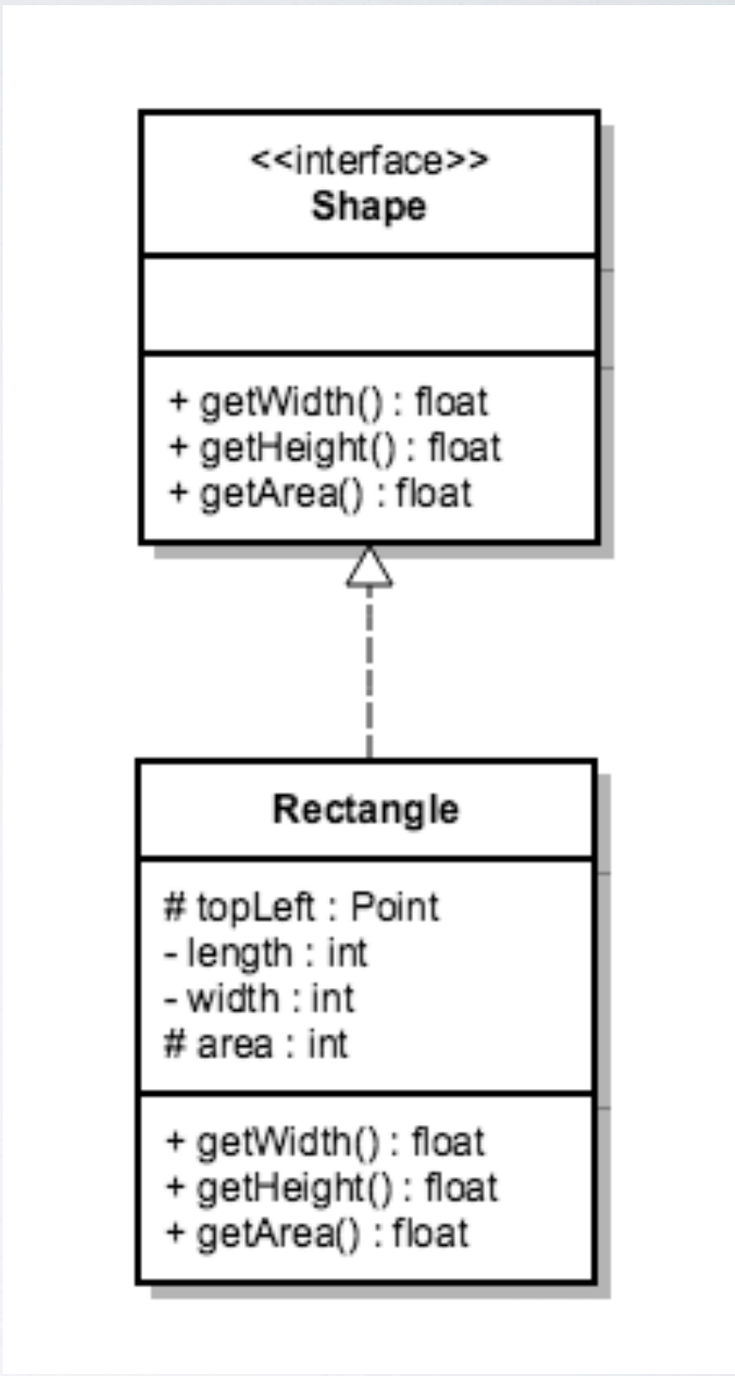
ABSTRACT CLASS

- Abstract classes are not instantiable
- Abstract methods do not have an implementation

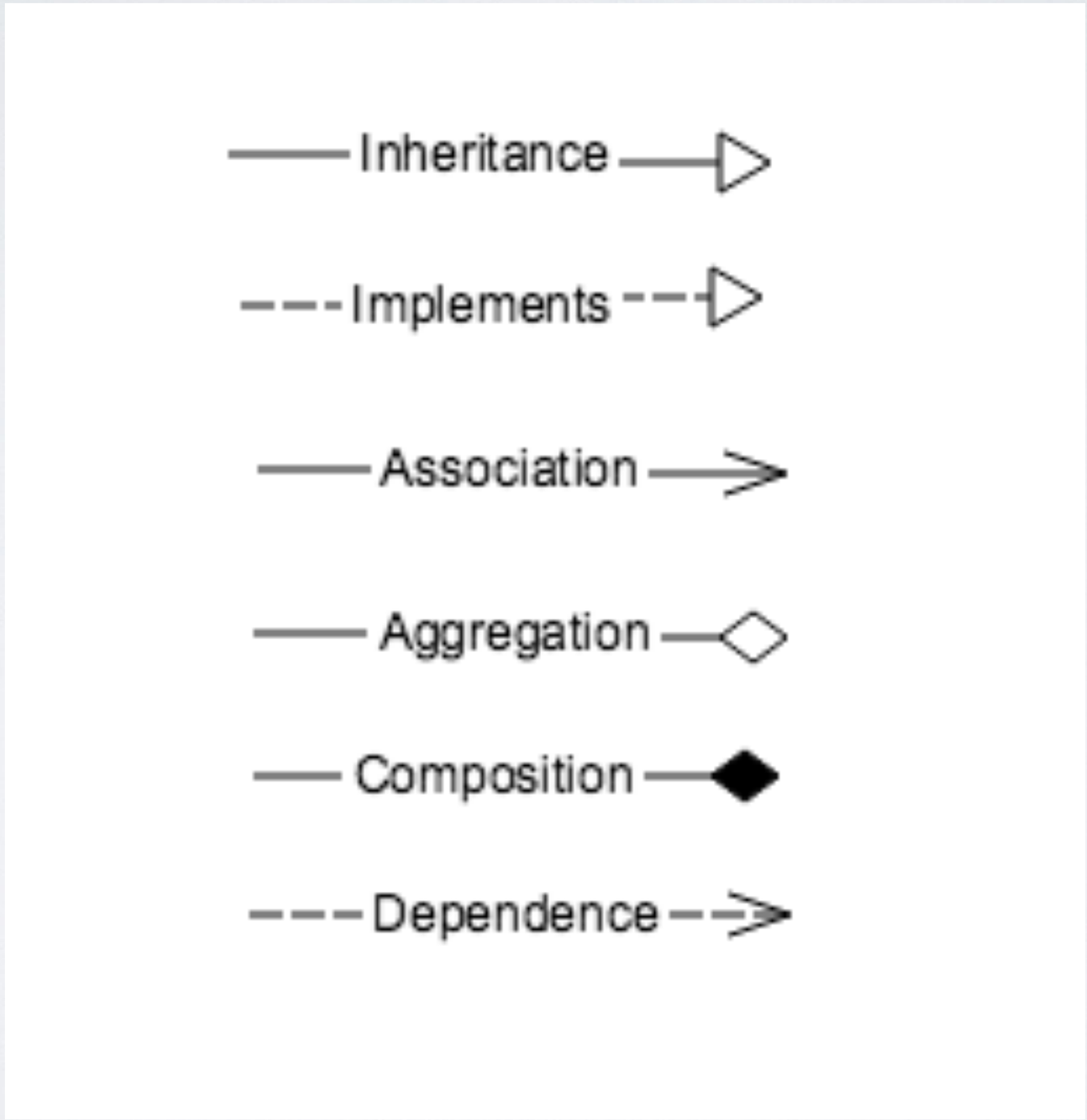


INTERFACE

- Think of an interface as a contract between a class and the outside world
- This contract is enforced by the compiler
- All methods defined by that interface must be implemented by any class implementing the interface
- Java Iterator: hasNext(), next(), remove()

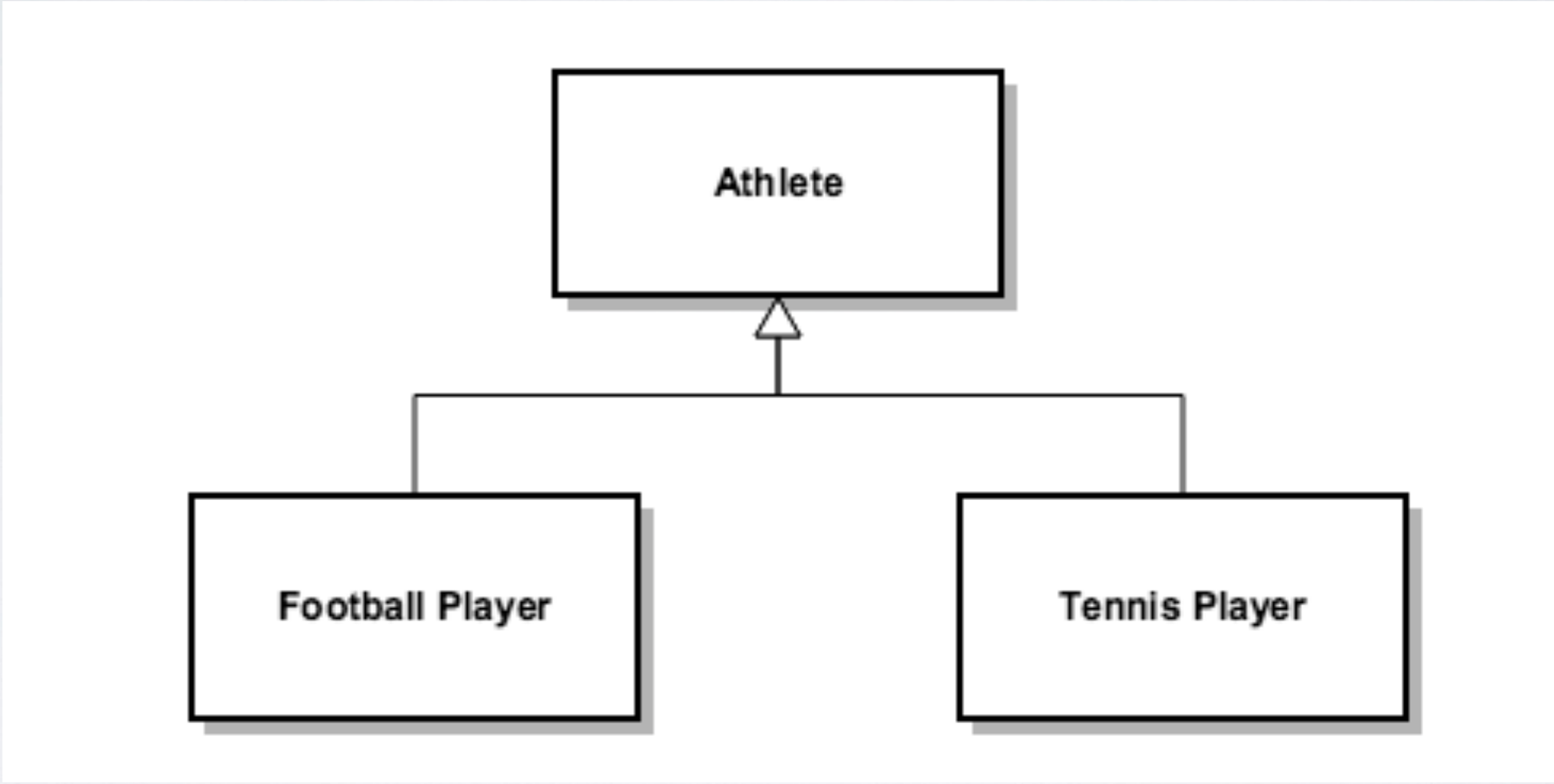


ARROWS


Inheritance
Implements
Association
Aggregation
Composition
Dependence

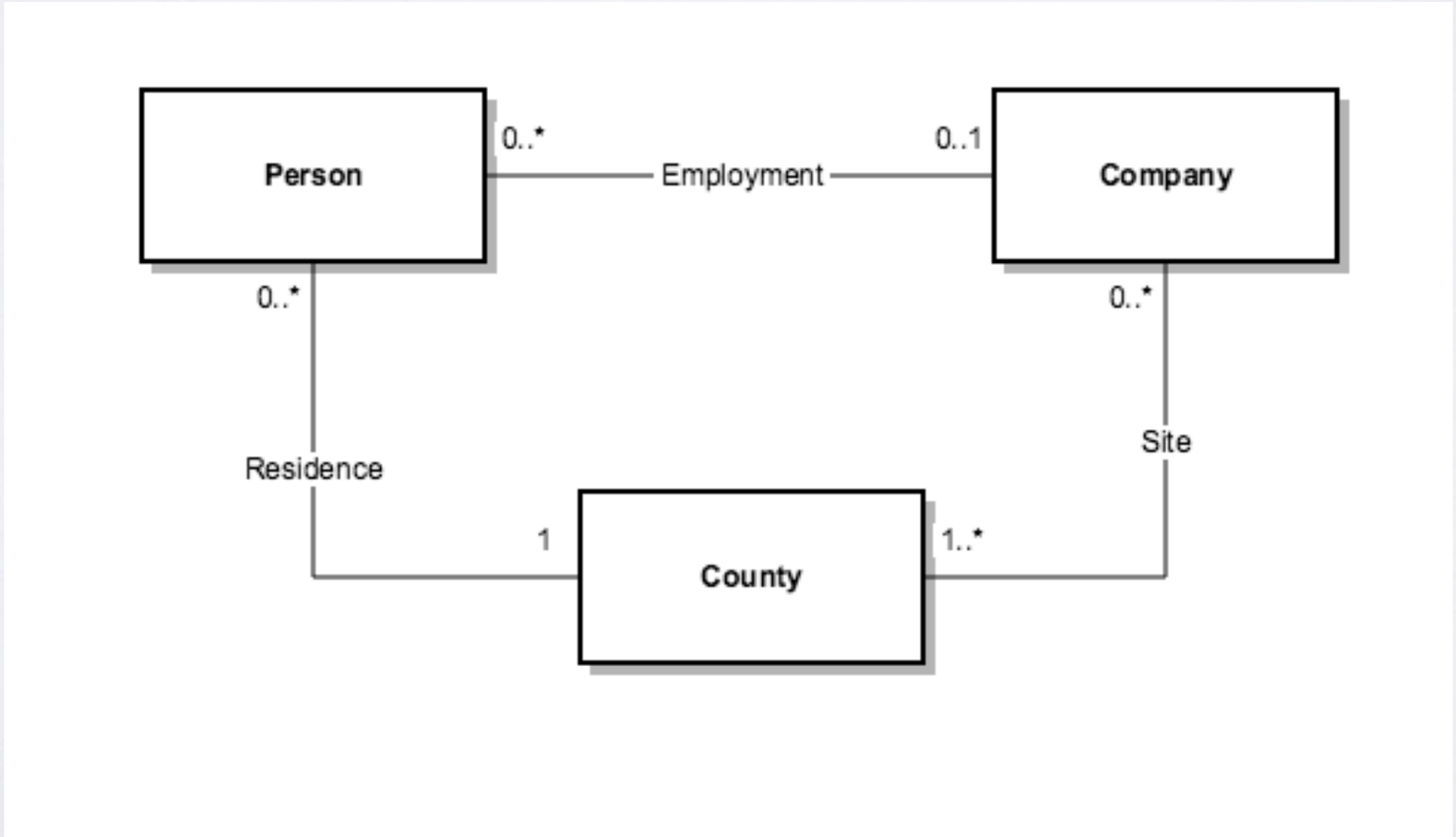
GENERALIZATION

Inheritance

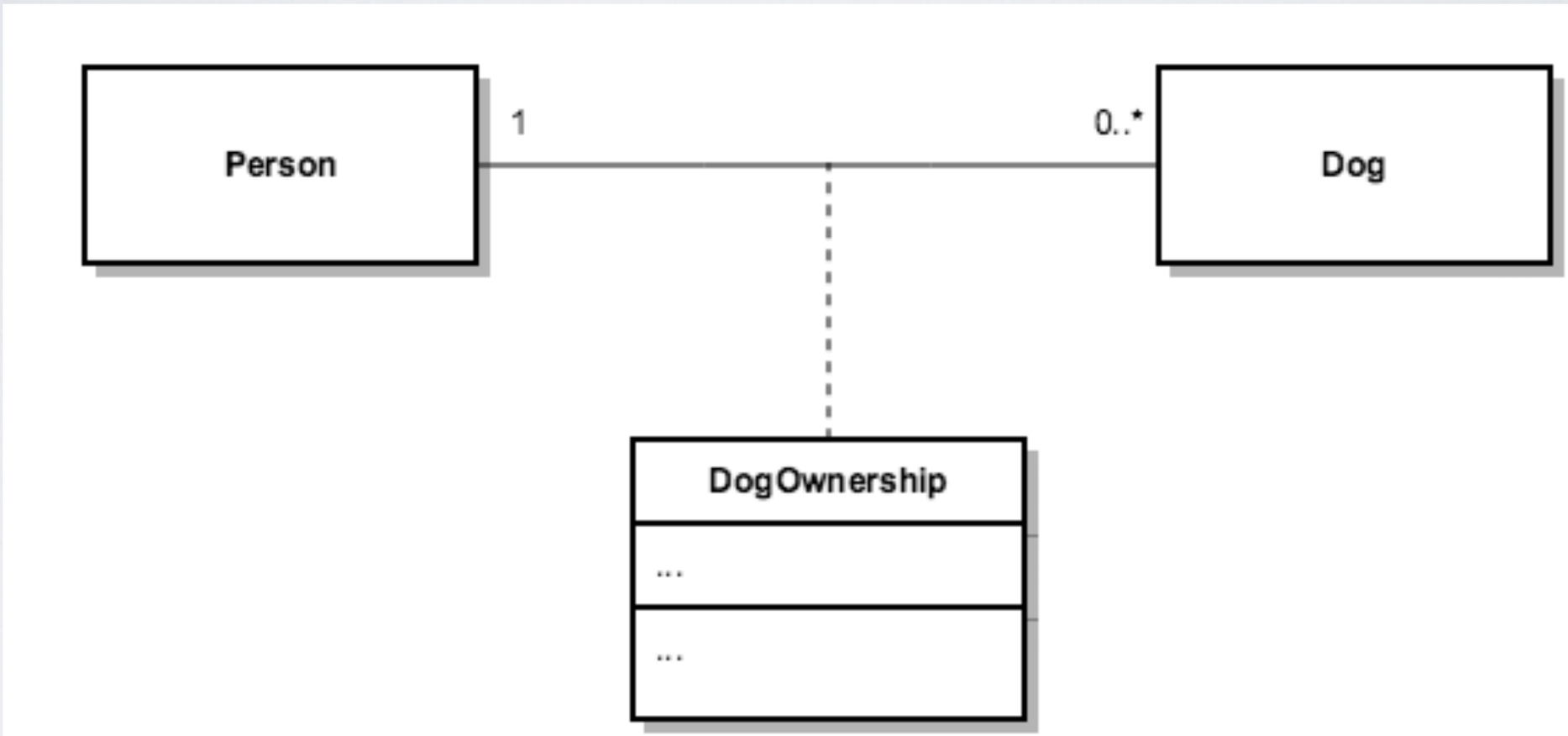


ASSOCIATION

- Used to model the relationship between classes
- By default, they are bi-directional (both classes are aware of each other)



ASSOCIATION CLASS



ASSOCIATION NAVIGATION

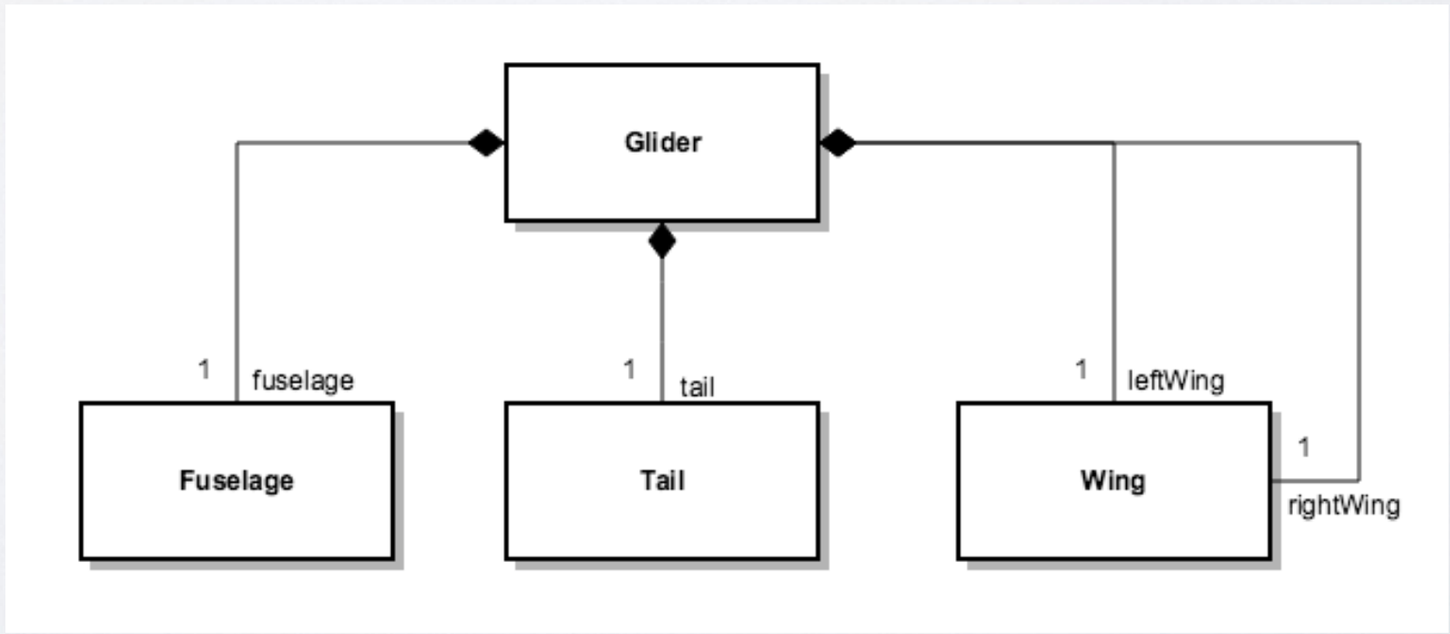
- Person knows about Dog

- Dog knows about Person

- Person and Dog know about each other (default)

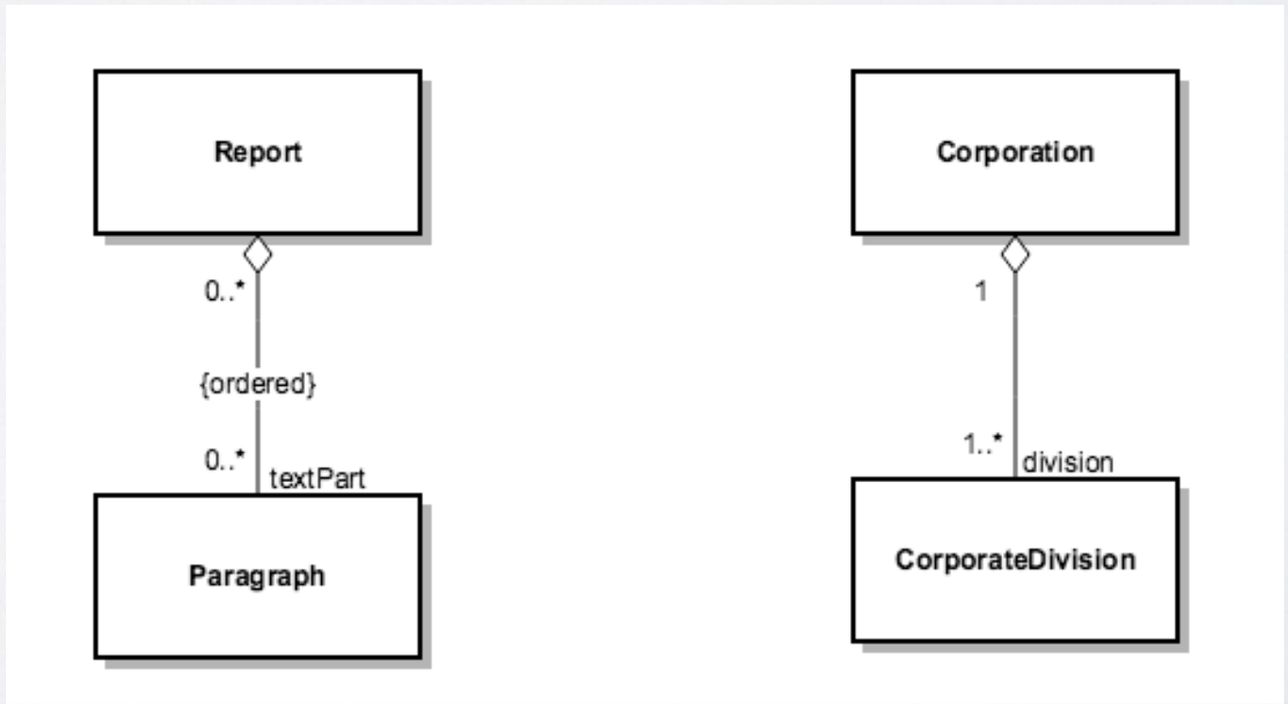
COMPOSITION

- Composite (whole) object does not exist without its components (parts)
- At any time, a component can only be part of one composite
- Composition is heteromeric (components will most likely be different)



AGGREGATION

- Aggregate (whole) may exist without constituents (parts)
- At any time, a constituent may belong to more than one aggregate
- Aggregation tends to be homeomeric (constituents belong to the same type)



COMPOSITION DISCUSSION

- When is it appropriate to model with composition? For example, why not use composition to show that a dog is composed of height, weight, color, and date of birth?

- A characteristic of composition is cascading delete. Can you think of a situation in which you'd want to delete a composite object but retain the component objects?

COMPOSITION DISCUSSION

- A sliced loaf is made of bread slices. Is the association between the loaf and its slices composition or aggregation?

- Object-portion composition

WAREHOUSE EXAMPLE

- Company XYZ is a manufacturing company that produces cartoon action figures for entertainment companies

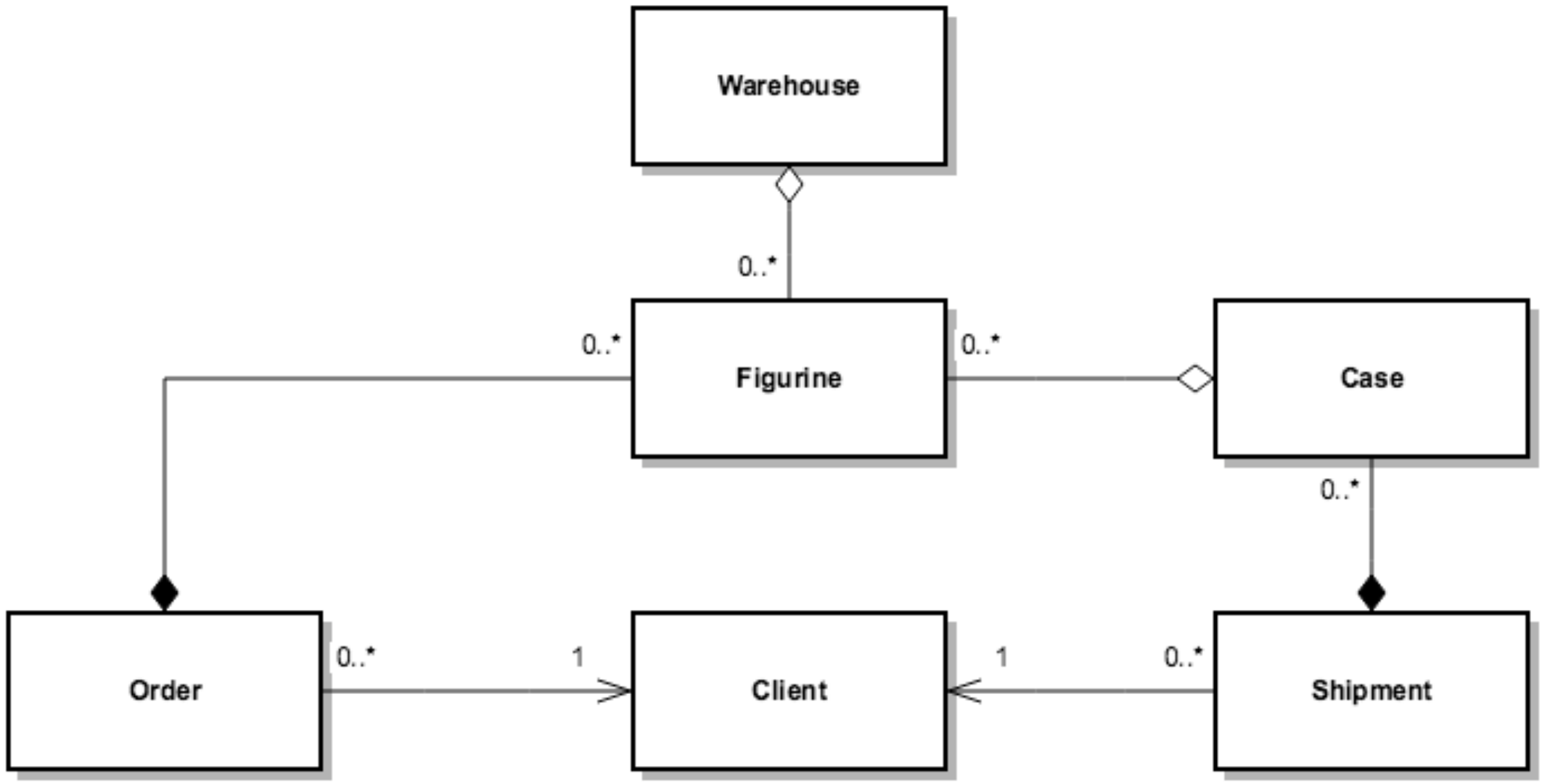
- This company needs an inventory and tracking system

- The inventory system keeps track of how many of each figurine is stored in each warehouse.

- Figures are stored in cases.

- Clients order the figurines and the cases are eventually shipped to clients.

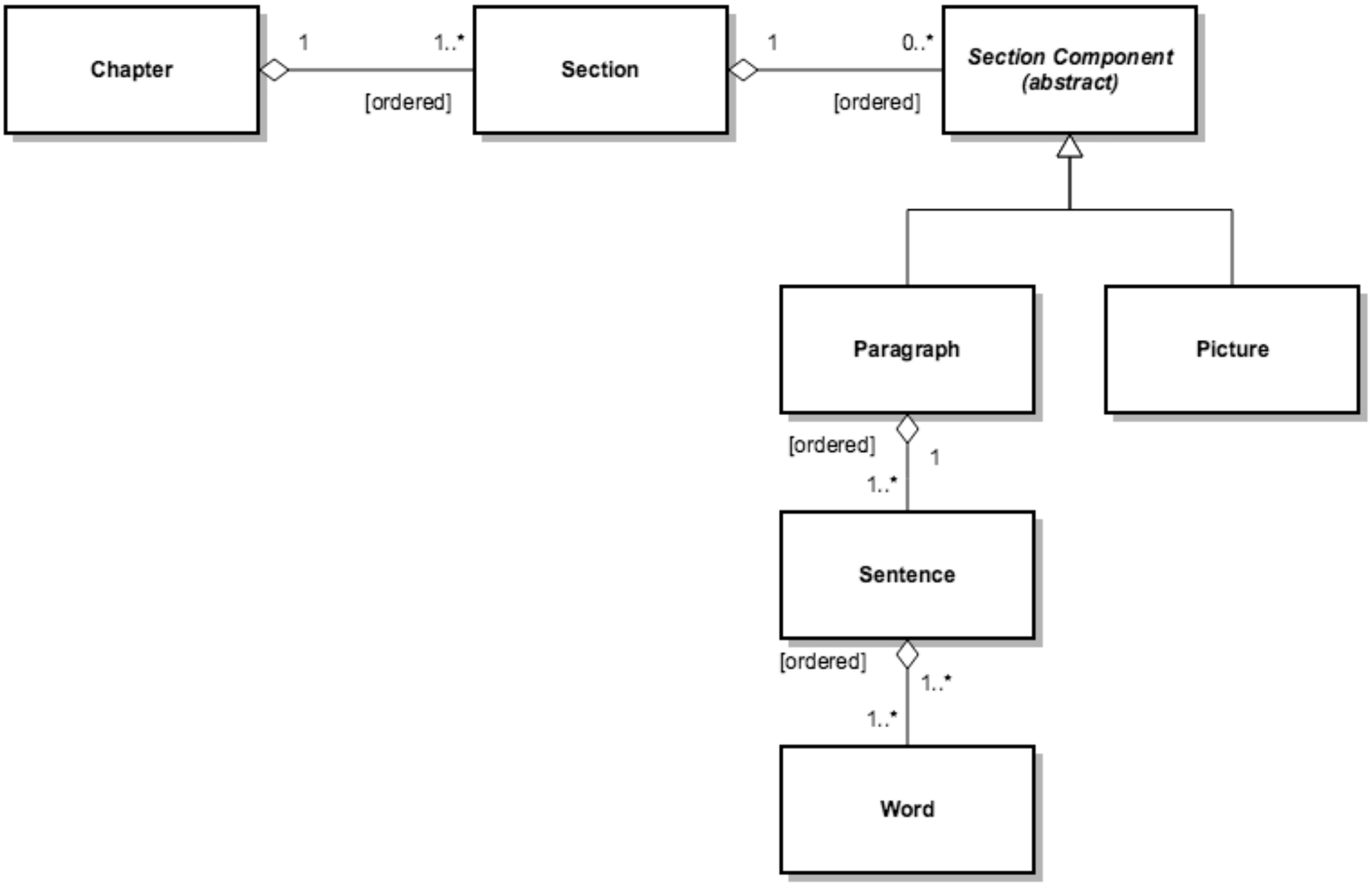
WAREHOUSE SOLUTION



BOOK CHAPTER EXAMPLE

- A chapter comprises several sections, each of which comprises several paragraphs and figures.
- A paragraph comprises several sentences, each of which comprises several words.

BOOK CHAPTER SOLUTION



FOR NEXT TIME…

- Read Chapter 3 from the Design of Everyday Things